P. Aaron Lott

AMSC 662

Project Proposal

Performance analysis of the 1GHz Motorola G4 RISC processor versus the 1.13 GHz P3 and 2.4 GHz P4 Intel CISC processors

Motivation

In 1991, IBM, Apple Computer and Motorola began a PowerPC alliance and began shipping computers with PowerPC RISC chips by 1994 to compete against Intel's CISC based chips. Traditionally RISC based chips run at a slower clock speed but have the ability to process several instructions in one clock cycle. At the current time, Motorola produces a 1GHz Risc processor that I have access to it is known as the G4. The G4 has 2 very nice characteristics from a scientific computing standpoint. First it has the ability to compute two 32bit floating point numbers in a single clock cycle or one double precision floating point calculation in a clock cycle, whereas Intel can only compute one single precision floating point number in a clock cycle and takes 2 clock cycles to compute a double precision floating point number. The second feature that strikes those interested in scientific computation is the G4's 2MB level 3 cache. This cache is primarily used for instruction misses, data load or store misses and cache management. With such a combination of resources, one may think to ask how the G4 compares to Intel's CISC based chips in floating point arithmetic as well as matrix/vector operations. The SPEC benchmark, which performs numerous integer and floating point operations has been performed on both a 1GHz G4 and a 1GHz G3. SPEC ranks the 1GHz G4 almost identical to 1GHz P3 on integer calculations, but against the what we have discussed above, the G4 ranks 2 times slower than the P3. To this end, I would like to propose to test the G4 processor against both a 1GHz Pentium III chip and a 2.4 GHz Pentium IV chip, in floating point and integer arithmetic as well as matrix vector operations. The goal of this project is to understand the G4 chip set so that one can take advantage of both double precision arithmetic and the large level 3 cache. I hope in the end that the code written for this project can be used to find and optimize architecture dependent resources and constraints so that one can take full advantage of the architecture one has to run their code on, be it PowerPC or Intel.

Method

I will begin by testing the efficiency of basic operations such as addition and multiplication. To do this I will use the code from section 9.3 in the text which allows me access the Intel cycle counter. To test the number of cycles these operations take on an Intel based machine. I will then write assembly code for the G4 chip to access its cycle counter and perform a similar analysis on that chip. Once I have an understanding of the cycles per operation. I will write loops to perform this operation over large vectors and compute the CPE as done in the class notes to try to gain optimal performance for each of the three architectures. I will then perform a matrix transpose operation, and a SAXPY matrix operation to test/optimize these either using blas or perhaps some other respected matrix vector library. Finally, I will use the time.h library to time these routines to see the actual speedup in code performance.

Validation

Although the SPEC benchmarks rank the G4 2 times worse than the P3, we hope to be able measure the 1GHz G4 processors floating point performance to be between the 1.13 GHz P3 and the 2.4 GHz P4 in our calculations. If, however, we cannot find such improvements due to the findings of our cycle counter analysis we hope to find the specific reason why the G4 is slower than the P3 in floating point operations. One item that may come into consideration is the factor of the latency of floating point operations on the G4 compared to the P3.